UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/005,248	12/03/2001	Brian C. Barnes	2000.056500	7937

23720 7590 08/23/2007
WILLIAMS, MORGAN & AMERSON
10333 RICHMOND, SUITE 1100
HOUSTON, TX 77042

EXAMINER	
LEMMA, SAMSON B	
ART UNIT	**PAPER NUMBER**
2132	

MAIL DATE	DELIVERY MODE
08/23/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Application Number: 10/005,248
Filing Date: December 03, 2001
Appellant(s): BARNES ET AL.

Jaison C. John
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed on April 05,2007

appealing from the Office action mailed August 24, 2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

2) Related Appeals and Interferences

The following are the related appeals, interferences, and judicial proceedings known to the examiner which may be related to, directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal:

On October 18, 2005, this application was given a nonstatutory obviousness-type double patenting rejection as being unpatentable over claims of the copending Application No. 10/005225.The other application/Application No. 10/005225, which was examined by another examiner and the present application were rejected with one and the same primary reference namely Dravis. Both applications have similar limitation except application no. 10/005,225 has extra limitation in its independent claims. Appellant on 08/05/2005 filed an appeal brief, appealing the final rejection given to the application No. 10/005,225 however the argument was not found persuasive and the board on 03/16/2007 affirmed examiner's rejection.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

- **Claims 1-3, 7-11, 15-19 and 23-24** stand rejected under 35 U.S.C. 102 as being anticipated by **Draves** (U.S. Patent No 5,802,590).
- **Claims 4-6, 12-14 and 20-22** stands rejected under 35 U.S.C. 103 as being unpatentable over **Draves** (U.S. Patent No 5,802,590)**in view of Krueger**(U.S. Patent No 4,962,533).

Furthermore,

- **Claims 1-3, 7-11, 15-19 and 23-24** stand rejected under 35 U.S.C. 102 as being anticipated by **Kamiya** (U.S. Patent No 4,949,238).
- **Claims 4-6, 12-14 and 20-22** stands rejected under 35 U.S.C. 103 as being unpatentable over **Kamiya** (U.S. Patent No 4,949,238) **in view of Krueger**(U.S. Patent No 4,962,533).

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

5,802,590	**Draves**	09-1998
4,949,238	**Kamiya**	08-1990
4,962,533	**Krueger**	10-1990

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

1. **Claims 1-3, 7-11,15-19 and 23-24** are rejected under 35 U.S.C. 102(b) as being anticipated by **Richard P. Draves** (hereinafter referred as **Draves**) (U.S. Patent No 5,802,590)

2. **As per claim 9 Draves discloses an apparatus, comprising:**

- **A processor for running code thereon**,[column 3, lines 39-42 and column 1, lines 11-22 and figure 2, ref. Num "250"] (As indicated on column 3, lines 39-42, the invention is directed towards a method and system in a kernel of an operating system for providing secure access to computer system resources. The OS kernel is inherently operates in the processor. And as it is indicated on column 1, lines 39-42, the portion of the operating system that is responsible for the allocation and deallocation of resources is known as the kernel. The kernel interacts with the shell and other programs as well as with the hardware devices on the system, including the processor (also called the central processing unit or CPU), memory and disk drives.)

- **For associating a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by the processor;** [column 3, lines 43-50 and column 3, lines 60-62] (As it is disclosed on column 3, lines 60-62, each process which is defined as concurrently executing computer programs on column 1, lines 14-15, meets the limitation each of a plurality of instructions or a set of instructions are inherently executed by the processor are associated with the resource identifier comprising the handle/key pair that is passed to the process/programs/set of instructions when requesting allocation of resources. Furthermore Draves on column 3, lines 43-50 discloses the following. In a preferred embodiment, the kernel maintains a system-wide resource table that is a hash table and that contains a resource entry corresponding **to each resource allocated by the kernel. The allocated resources are identified by a kernel-generated resource identifier.** The system of the present invention uses resource identifiers that contain both a handle and a key (a handle.backslash.key pair). The key is a very large number (e.g., 128 bits) that uniquely identifies the resource) **Wherein**

- **The processor receives** [column 3, lines 63-65; The OS kernel is inherently operates in the processor] **a request to execute at least one of the plurality of instructions or set of instructions by the code running thereon obtains a second security ID associated with the code,** [column 3, lines 60- 62 and column 3, lines 39-41] (As it is disclosed on column 3, lines 60-62, each process requesting the allocation of resource which is defined as concurrently executing computer programs on column 1, lines 14-15, meets the limitation, a request to execute at least one of the plurality of instructions or set of instructions by the code running thereon obtains resource identifier comprising the handle/key pair that is uniquely identify the resources as explained on column 49-51 **meets the limitation of obtaining a second security ID associated with the process/program/code)**

- **Compares the second security ID with the first security ID, and executes the requested instruction or set of instructions providing that the second security ID matches the first security ID. [Column 4, lines 8-10]** (When a matching key is found, the kernel allows the process to access/executes the requested instruction or set of instructions /resource/program as explained on column 4, lines 8-10)

3. **As per claim 1,** Claim 1 recites the method version of the independent claim 9 and likewise rejected by the same analogy/ground as that of claim 9.

4. **As per claim 17,** Claim 17 recites the same limitations as that of the independent claim 9 and therefore rejected by the same analogy/ground as that of claim 9.

5. **As per claims 2, 10 and 18, Draves** discloses the method/apparatus/article as applied to claims 1, 9 and 17 above. Furthermore, **Draves** discloses the method/apparatus/article comprising denying the execution of the requested instruction or set of instructions providing that the first and second security IDs mismatch. [Column 4, lines 5-8; figure 8, ref. Num "830"]

6. **As per claims 3, 11 and 19, Draves** discloses the method/apparatus/article as applied to claims 1, 9 and 17 above. Furthermore, **Draves** discloses the method/apparatus/article wherein associating a first security identification (ID) further comprises: storing a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by a processor.[Column 3, lines 59-62] (The stored resource contains a copy of the key meets the recitation of this claim.)

7. **As per claims 7, 15 and 23, Draves** discloses the method/apparatus/article as applied to claims 1, 9 and 17 above. Furthermore, **Draves** discloses the method/apparatus/article wherein comparing the second security ID with the first security ID further comprises:

comparing a portion of the second security ID with a portion of the first security ID.[Column 4, lines 8-10] (A process access for executing the requested instruction or set of instructions or a program or in general accessing the resource is allowed when a match is found by comparing all portions of the first and second identification)

8. **As per claims 8, 16 and 24, Draves** discloses the method/apparatus/article as applied to claims 7, 15 and 23 above. Furthermore, **Draves** discloses the method/apparatus/article wherein executing the requested instruction or set of instructions providing that the second secuity ID matches the first security ID further comprises:

- Executing the requested instruction or set of instructions providing that the portion of the second security ID matches the portion of the first security ID. [Column 4, lines 8-10] (A process access for executing the requested instruction or set of instructions or a program or in general accessing the resource is allowed when a match is found by comparing all portions of the first and second identification)

Claim Rejections - 35 USC § 103

9 **Claims 4-6, 12-14 and 20-22** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Richard P. Draves** (hereinafter referred as **Draves**) (U.S. Patent No 5,802, 590) in view of **Krueger et al**, (hereinafter referred to as **Krueger**) (U.S. Patent No. 4,962,533)

10 **As per claims 4-6, 12-14 and 20-22, Draves discloses**

- **A processor** [Figure 2, ref. Num "250"] **for running code thereon**,[Column 1, lines 13-14; column 4, lines 16-17] and **for associating a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by the processor;** [Figure 3, ref. "handle/key"] (As shown on figure 3, for each multiplicity/plurality of processes a handle/key pair is associated.)

Draves does not explicitly discloses

- A first security identification (ID) further comprises:

Classifying at least one instruction or set of instructions from a plurality of instructions that are to be executed by a processor as being security sensitive;
And associating a first security identification (ID) with each of the instructions or set of instructions that are classified as security sensitive.

However, in the same field of endeavor, **Krueger** discloses

Classifying at least one instruction or set of instructions from a plurality of instructions that are to be executed by a processor as being security sensitive and associating a first security identification (ID) with each of the instructions or set of instructions that are classified as security sensitive;[Column 2, lines 43-46; abstract and] (computer system uses security labels for every word in memory and according to the present invention, in a computer system every word in the memory has a corresponding label/security identification. This label indicates the security classification, and compartments if any, of that word of data)

It would have been obvious to one having ordinary skill in the art, at the time the invention was made, to combine the features of having a classification for one instructions/program and associating security identification/label with each instructions or set of instructions as per teachings of **Krueger** in to the method as taught by **Draves**, in order provide a security technique for a computer system in which data retains its classification with a straightforward and reliable mechanism for separating sensitive and non-sensitive data within the system.[see **Krueger** column 2, lines 19-21 and 39-41]

11 The indicated allowability of claims 1-24 is also withdrawn in view of the newly discovered reference(s) to **Kamiya**, Shigeo (hereinafter referred to as **Kamiya**) (U.S. Patent No. 4, 949, 238)

12 **Claims 1-3, 7-11,15-19 and 23-24** are rejected under 35 U.S.C. 102(b) as being anticipated by **Kamiya**, Shigeo (hereinafter referred to as **Kamiya**) (U.S. Patent No. 4, 949, 238)

13 As per claims **Claims 1-3, 7-11,15-19 and 23-24**, **Kamiya** discloses **a method, comprising**

- **Associating a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by a processor** [Column 2, line 67-column 3, line 10; column 4, lines 49-68; column 5, lines 26-27 and figure 1].(the plurality of branch instructions meets the limitation of plurality of instruction as it is disclosed on column 2, line 67-column 3, line 10. And the true/mask register shown on figure 1, ref. Num "122" meets the limitation of the first security ID.)

- **Requesting to execute at least one of the plurality of instructions of set of instructions by a software code running on the processor;** [Column 5, lines 23-25) (branch instruction executed .....)

- **obtaining a second security ID associated with the software code running on the processor;** (column 3, line 41-42, "the current privilege register")

- **comparing the second security with the first security ID ; [column 3, lines 35-42] and**

- **Executing the requested instruction or set of instructions** [column 2, lines 22-24, "the succeeding microinstruction is normally selected"] **providing that the second security ID matches the first security ID.[column 2, lines 22-24,** ("determined to be true" meets the limitation of the second security ID matches the first security ID)

14 **Claims 4-6, 12-14 and 20-22** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Kamiya**, Shigeo (hereinafter referred to as **Kamiya**) (U.S. Patent No. 4, 949, 238) in view of **Krueger et al**, (hereinafter referred to as **Krueger**) (U.S. Patent No. 4,962,533)

15 **As per Claims 4-6, 12-14 and 20-22 Kamiya** discloses **a method, comprising**

- **Associating a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by a processor** [Column 2, line 67-column 3, line 10; column 4, lines 49-68; column 5, lines 26-27 and figure 1].(the plurality of branch instructions meets the limitation of plurality of instruction as it is disclosed on column 2, line 67-column 3, line 10. And the true/mask register shown on figure 1, ref. Num "122" meets the limitation of the first security ID.)

- **Requesting to execute at least one of the plurality of instructions of set of instructions by a software code running on the processor; [**Column 5, lines 23-25) (branch instruction executed .....)

- **obtaining a second security ID associated with the software code running on the processor;** (column 3, line 41-42, "the current privilege register")

- **comparing the second security with the first security ID ; [column 3, lines 35-42] and**

- **Executing the requested instruction or set of instructions [**column 2, lines 22-24, "the succeeding microinstruction is normally selected"] **providing that the second security ID matches the first security ID.[column 2, lines 22-24, (**"determined to be true" meets the limitation of the second security ID matches the first security ID)

Kamiya does not explicitly discloses

- A first security identification (ID) further comprises:

Classifying at least one instruction or set of instructions from a plurality of instructions that are to be executed by a processor as being security sensitive;

And associating a first security identification (ID) with each of the instructions or set of instructions that are classified as security sensitive.

However, in the same field of endeavor, **Krueger** discloses

Classifying at least one instruction or set of instructions from a plurality of instructions that are to be executed by a processor as being security sensitive and associating a first security identification (ID) with each of the instructions or set of instructions that are classified as security sensitive;[Column 2, lines 43-46; abstract and] (computer system uses security labels for every word in memory and according to the present invention, in a computer system every word in the memory has a corresponding label/security identification. This label indicates the security classification, and compartments if any, of that word of data)

It would have been obvious to one having ordinary skill in the art, at the time the invention was made, to combine the features of having a classification for one instructions/program and associating security identification/label with each instructions or set of instructions as per teachings of **Krueger** in to the method as taught by **Kamiya**, in order provide a security technique for a computer system in which data retains its classification with a straightforward and reliable mechanism for separating sensitive and non-sensitive data within the system.[see **Krueger** column 2, lines 19-21 and 39-41]

(10) Response to Argument

Appellant's argument filed with the Appeal brief, on April 05, 2007 have been fully considered but they are not persuasive.

Before responding to the Appellant's argument, Examiner would point out how each and every limitation of the independent claims, namely independent claims 1, 9 and 17 are disclosed by the reference/s on the record namely **Draves**.

In a nutshell, the independent claims 1, 9 and 17 and the invention as it is disclosed on figure 5 of Appellant's disclosure performs the following steps.

1. Assigning a security Identification ID for security sensitive instructions and storing them in the register/Address Table. Appellant's independent claims designate this as "**first Security ID**".

2. When software code which is currently running on the processor desires to execute the security sensitive instructions, its security ID is obtained by referencing address of the memory table, which Appellant's independent claims designates this as **"Second Security ID"**.

3. Then If the software's code security ID which is Security ID of the code running on the processor which is designated as **"second security ID"** is the same as the security ID of the requested security sensitive instructions designated as the "**first security ID**", then execution of the security sensitive instruction by the software code is granted otherwise it is denied.

In view of the above understanding, Examiner would point out that each and every limitation of the independent claims 1, 9 and 17 is anticipated by the reference on the record, namely Draves.

For instance referring to the independent claim 1,

Draves discloses a method[Column 1, lines 6-8] (*On column 1, lines 6-8, the following has been disclosed. , "The present invention relates to the field of computer systems, and more particularly, **to a method** and system for ensuring that only authorized processes access resources"*)

comprising:

- **Associating a first security identification (ID) with each of a plurality of instructions or a set of instructions that are to be executed by a processor;** [Column 1, lines 33-34] (*The kernel returns to the spreadsheet program a resource identifier. This resource identifier meets the limitation of first security identification (ID). And spreadsheet program is an application program and is a plurality of instructions or a set of instructions. And on column 1, lines 11-15, it has been disclosed that concurrently executing computer program is managed by the operating system and is referred as a process. Furthermore, on column 1, lines 23-24, this spreadsheet program is referred as a process. And it is known in the art that a process/computer program/set of instructions is executed by the processor.)*

- **Requesting to execute at least one of the plurality of instructions or set of instructions by a software code running on the processor; [Column 1, lines 35-41]** *(As it is been disclosed on column 1, lines 35-38, the kernel is requested so that the word processing program which meets the limitation of software code shares a block of memory with the Spread sheet program/plurality of instructions or set of instruction so that the word processing program/software code can access the spreadsheet data. Therefore, the request is made so that spread sheet program's data/plurality of instructions or set of instruction's data is accessed or executed by the word processing program/software code. And it is well known fact that application program such as word processing program/software codes runs on the processor.)*

- **Obtaining a second security ID associated with the software code running on the processor; [Column 1, lines 36-38 and column 1, lines 39-41]** *(the kernel generates **another resource identifier/second security ID** that the word processing program/software code can use so that spread sheet program's data/plurality of*

instructions or set of instruction's data is accessed or executed by the word processing program/software code. And it is well known fact that application program such as word processing program/software codes runs on the processor.)

- **Comparing the second security ID with the first security ID; and executing the requested instruction or set of instructions providing that the second security ID matches the first security ID. [Column 3, lines 60-column 4, lines 2; and column 4, lines 8-10]** *[On column 3, lines 60-column 4, lines 2, it has been disclosed that, the resource identifier comprising the handle/key pair and when a process wishes to access the allocated resources it passes the handle/key pair to the kernel. The kernel examines the resource entry indexed by the passed handle to determine whether the passed key is equal to the key in the indexed resource entry. The keys may not be equal for several reasons, including resource table compaction and attempted forgery. And column 4, lines 8-10, When a matching key is found, the kernel allows the process to access/executes the requested resource as explained on column 4, lines 8-10, this meets the limitation recited as* ***"comparing the* second security ID with the first security ID"***. *However to further clarify how the comparison is done, examiner would show the following.*

As it is shown above, Draves on column 1, lines 33-34 teaches, the following, "The kernel returns to the spreadsheet program a resource identifier. This resource identifier meets the limitation of first security identification (ID). And spreadsheet program, is an application program and is a plurality of instructions or a set of instructions. **Examiner further asserts that** *on column 1, lines 35-38, the kernel is requested so that the word processing program which meets the limitation of software code shares a block of memory with the Spread sheet program/plurality of instructions or set of instruction so that the word processing program/software code can access the spreadsheet data. Therefore, the*

request is made so that spread sheet program's data/plurality of instructions or set of instruction's data is accessed or executed by the word processing program/software code. And it is well known fact that application program such as word processing program/software codes runs on the processor.

Therefore combining the above teachings the following deduction is made by the office. When the word processing program/software code/process wishes to access the spread sheet program's data, it passes its handle/key pair/the another resource identifier/second security ID, the kernel examines the resource entry indexed by the passed handle to determine whether the passed key/second security ID is equal to the key indexed resource entry/the first security identifier. And keys/identifiers match, the kernel allows the word processing program/software code/process to access/executes the spread sheet program's data/set of instructions data and this meets the recitation of the claimed limitation.)

Note: Examiner would point out that, the above mapping of the reference/s on the record to the argued independent claims, would not only clarify examiner's interpretations but also apply/reply to almost all of the Appellant's argument However, having shown how the reference on the record is mapped to each and every limitation of the argued independent claims, Examiner would continue to respond to the Appellant's argument as follows.

Referring to the independent claims 1, 9 and 17, Appellant first argued that Draves, the reference on the record, does not describe or suggest the limitation "obtaining a second security ID of the software code currently being executed by the processor" **Examiner disagrees with this argument**.

Examiner would point Dravis on column 1, lines 36-38 and column 1, lines 39-41 discloses that the kernel **generates another resource identifier/**second security ID that the word processing program/software code can use so that spread sheet

program's data/plurality of instructions or set of instruction's data is accessed or executed by the word processing program/software code. And it is well known fact that application program such as word processing program/software codes runs on the processor.

Furthermore,

Examiner would point out that Dravis on column 2, lines 27-31, discloses, the following. "The system provides for ensuring that a computer program is authorized to access a computer system resource. The system generates a system-wide resource table that has a resource entry **for each allocated resource. Each resource entry contains a preferably non-forgeable key that uniquely identifies the resource**. This indicates the fact that **non-forgeable are** associated **with several resources, in such a way that, and each resources/items** are also uniquely identified by the non-forgeable keys.

Examiner would also asserts that Dravis on column 3, lines 42-48, discloses the following, "In a preferred embodiment, the kernel maintains a system-wide resource table that is a hash table and that contains a resource entry corresponding **to each resource allocated by the kernel. The allocated resources are identified by a kernel-generated resource identifier.** The system of the present invention uses resource identifiers that contain both a handle and a key (a handle.backslash.key pair)." This also indicates the fact there is also several resources/items which are identified by the resource identifier or key pair.

Dravis further discloses the following, "When a process wishes to access the allocated resource, it passes the handle.backslash.key pair to the kernel. The kernel examines the resource entry indexed by the passed handle to determine whether the passed key is equal to the key in the indexed resource entry. The keys may not be equal for several reasons, including resource table compaction **and attempted forgery**." [Column 3, lines 63-column 4, line 2]. This implies that the requesting process could be any

process including an unauthorized process which is attempting forgery however forgery process is not able to access other resource that it is not authorized since it does not have the right key pair and the kernel denies this process from accessing the resources by matching the key with the resource it is requesting.

Secondly, Appellants argued that the limitation recited as, "executing the requested instruction or set of instructions providing that the second security ID associated with the software code running on the processor matches the first security ID" is not disclosed by the reference/Dravis.

Examiner disagrees with this argument.

Examiner would point on column 3, lines 60-column 4, lines 2, Dravis discloses that, the resource identifier comprising the handle/key pair and when a process wishes to access the allocated resources it passes the handle/key pair to the kernel. The kernel examines the resource entry indexed by the passed handle to determine whether the passed key is equal to the key in the indexed resource entry. The keys may not be equal for several reasons, including resource table compaction and attempted forgery. And column 4, lines 8-10, When a matching key is found, the kernel allows the process to access/executes the requested resource as explained on column 4, lines 8-10. Finally Dravis on column 1, lines 36-38 and column 1, lines 39-41 discloses that the kernel generates another resource identifier/second security ID that the word processing program/software code can use so that spread sheet program's data/plurality of instructions or set of instruction's data is accessed or executed by the word processing program/software code. And it is well known fact that application program such as word processing program/software codes runs on the processor.

Therefore combining the above teachings, the following deduction is made by the office. When the word processing program/software code/process wishes to access the spread sheet program's data, it passes its handle/key pair/another resource identifier/second

security ID, and the kernel examines the resource entry indexed by the passed handle to determine whether the passed key/second security ID is equal to the key indexed resource entry/the first security identifier. And if the keys/identifiers match, the kernel allows the word processing program/software code/process to access/executes the spread sheet program's/set of instructions data.

Thirdly, Appellants argued that the examiner uses the same "process" in Draves to satisfy the requirements of "instruction" and another distinct requirement of the "software code" that execute the "process", ie,. Instruction(s), as set forth in claim 1.

Examiner disagrees with this argument.

As it is shown above, **spreadsheet program is an application program and is interpreted as a plurality of instructions or a set of instructions**. And on column 1, lines 11-15, it has been disclosed that concurrently executing computer program is managed by the operating system and is referred as a process. Furthermore, on column 1, lines 23-24, this spreadsheet program is referred as a process. On the other side, as it is been disclosed on column 1, lines 35-38, the kernel is requested so that the **word processing program which meets the limitation of software code** shares a block of memory with the Spread sheet program/plurality of instructions or set of instruction so that the word processing program/software code can access the spreadsheet data. Therefore, the request is made so that spread sheet program's data/plurality of instructions or set of instruction's data is accessed or executed by the word processing program/software code.

T**herefore, contrary to Appellant's argument,** the distinct requirement of the two terms "instruction" and "software code" is satisfied by dravis's reference. However Examiner again would point out that both are software/instructions.

Referring to the dependent claims 4-6, 12-14 and 20-22, Appellant's argued that the cited references fail to provide any suggestion or motivation for modifying the prior art to arrive Appellant's claimed invention.

Examiner disagrees and in response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, the motivation to combine the reference is explicitly disclosed by the secondary reference, namely Krueger on column 2, lines 19-21 and 39-41. Therefore as provided in the final office action of 08/24/2006 and this examiner's answer Krueger in fact provides motivation for modifying the prior art to arrive Appellant's claimed invention.

Note: On October 18, 2005, this application was given a nonstatutory obviousness-type double patenting rejection as being unpatentable over claims of the copending Application No. 10/005225.

The other application/Application No. 10/005225, which was examined by another examiner and the present application were rejected with one and the same primary reference namely Dravis. Both applications have similar limitation except application no. 10/005,225 has extra limitation in its independent claims.

Appellant on 08/05/2005 filed an appeal brief, appealing the final rejection given to the application No. 10/005,225 ***however the argument was not found persuasive and the board on 03/16/2007 affirmed examiner's rejection.***

11) Related Proceeding(s) Appendix

The Board's decision for application No. 10/005,225 is attached.

Respectfully submitted,

Samson Lemma.

GILBERTO BARRÓN JR
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Conferees:

Gilberto Barron

Lanier Benjamin E

11) Related proceedings Appendix

Board Decision for 10/005,225 (March 16, 2007)

The opinion in support of the decision being entered today was not written for publication and is not binding precedent of the Board

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Ex parte BRIAN C. BARNES, RODNEY SCHMIDT, and GEOFFREY STRONGIN

Appeal No. 2007-0564
Application 10/005,225

On Brief

Before LEE, TORCZON, and MOORE, Administrative Patent Judges.

LEE, Administrative Patent Judge.

DECISION ON APPEAL

A. Statement of the Case

This is a decision on appeal by an applicant under 35 U.S.C. § 134(a) from a rejection of claims 1-24 in application 10/005,225 ('225 application). We have jurisdiction under 35 U.S.C. § 6(b). The real party in interest is Advanced Micro Devices, Inc.

References relied on by the Examiner

Draves	U.S. Patent 5,802,590	September 1, 1998
Clifton	U.S. Patent 5,469,556	November 21, 1995
Holtey et al. (Holtey)	U.S. Patent 4,290,104	September 15, 1981

The Rejections on Appeal

1. The Examiner rejected claims 1-2, 6, 8-10, 14, 16-18, 22, and 24 as unpatentable for obviousness under 35 U.S.C. § 103 over the combined teachings of Draves and Clifton.

2. The Examiner rejected claims 3-5, 7, 11-13, 15, 19-21, and 23 as unpatentable for obviousness under 35 U.S.C. § 103 over the combined teachings of Draves, Clifton, and Holtey.

B. Issues

1. Has the applicant shown error in the obviousness rejection of claims 1-2, 6, 8-10, 14, 16-18, 22, and 24 over Draves and Clifton?

2. Has the applicant shown error in the obviousness rejection of claims 3-5, 7, 11-13, 15, 19-21, and 23 over Draves, Clifton, and Holtey?

C. Summary of the Decision

Applicant has not shown error in the obviousness rejection of claims 1-2, 6, 8-10, 14, 16-18, 22, and 24 over Draves and Clifton.

Applicant has not shown error in the obviousness rejection of claims 3-5, 7, 11-13, 15, 19-21, and 23 over Draves, Clifton, and Holtey.

D. Findings of Fact (Referenced as FF. ¶ No.)

1. The invention is directed to a method and apparatus for securing the operation of a computer system, by protecting against rogue programs which exploit security defects ('225 application, specification, page 2, lines 9-10; page 3, lines 12-21, page 4, lines 2-7).

2. It was a common security feature in a microprocessor environment to provide a hierarchy of privilege levels for software to be run on the computer, to differentiate and control the extent of authorized access for software ('225 application, specification, page 2, lines 20-23).

3. Typically, operating system programs and software drivers run at the highest privilege level, i.e., level 0, which permits free access to virtually all of the resources of the computer system, and application programs run at the lowest privilege level, i.e., level 3, which permits access to system resources usually only by permission of the operating system program ('225 application, page 2, line 23, through page 3, line 3).

4. Claims 1, 9 and 17 are the only independent claims and read as follows:

> 1. A method, comprising:
>
> associating a first security identification (ID) with each of a plurality of software codes that are to be executed by a processor at a most privileged level;
>
> initiating the execution of one of the plurality of software codes on the processor at the most privileged level;
>
> receiving a second security ID identifying a memory space that the one software code being executed on the processor is attempting to access;
>
> comparing the second security ID with the first security ID; and
>
> providing access to the memory space providing that the first and second security IDs match.
>
> 9. An apparatus, comprising:
>
> a processor for initiating the execution of one of a plurality of software codes at a most privileged level;
>
> a memory including a memory space for associating a first security identification (ID) with each of a plurality of software codes that are to be executed by the processor at the most privileged level;
>
> wherein the processor receives a second security ID identifying the memory space that the one software code being executed on the processor is attempting to access, compares the second security ID with the first security ID, and provides access to the memory space providing that the first and second security IDs match.
>
> 17. An apparatus, comprising:

means for associating a first security identification (ID) with each of a plurality of software codes that are to be executed by a processor at a most privileged level;

means for initiating the execution of one of the plurality of software codes on the processor at the most privileged level;

means for receiving a second security ID identifying a memory space that the one software code being executed on the processor is attempting to access;

means for comparing the second security ID with the first security ID; and means for providing access to the memory space providing that the first and second security IDs match.

5. Draves discloses a method and apparatus for allowing a process, i.e., software, to access system resources in a secured manner by first generating a "key" for a resource to be allocated to the process and sending to the process that key together with a handle which is an index pointer to a place within a resource table at which the generated key for an associated resource is stored. (Draves, Abstract, lines 1-11; column 2, lines 32-37; Figures 1, 6, and 7.) The process accesses the resource by submitting the handle/key pair and if the key matches a stored key for an allocated resource, then the process is allowed to access the resource. (Draves, Abstract, lines 11-16; column 2, lines 41-48; column 3, lines 63-64; Figure 9.)

6. In Draves's system, each resource potentially accessible by a process is associated with a non-forgeable key that uniquely identifies the resource. (Draves, column 2, lines 31-32.)

7. According to Draves' disclosure, only authorized processes are able to access resources. (Draves, column 4, lines 15-17.)

8. In a preferred embodiment of Draves, a process may share an authorized resource by passing the handle/key pair corresponding to the resource to another process with whom it is desirable to share the resource. (Draves, column 5, lines 34-36.) The second process,

called a client process (Draves, column 1, lines 53-57), may access the shared resource by submitting the handle/key pair. (Draves, column 5, lines 37-39.) The first process passing the handle/key pair for the resource to the client process is referred to as a server process. (Draves, column 1, lines 53-57; column 4, lines 61-62; column 7, lines 38-40.)

9. In the Background of the Invention portion of Draves, the discussion identifies a spreadsheet program and a word processing program as examples of two processes which may need to share a common resource. (Draves, column 1, lines 23-26.)

10. In the Detailed Description of Invention section and the Summary of Invention section of its disclosure, Draves refers to processes only generally and nowhere limits a "process" desiring access to a system resource to a spreadsheet program, a word processing program, or any other type of application program.

11. The reference to a spreadsheet program and a word processing program in the Background of the Invention portion of Draves merely identifies an example of programs which may need to share a resource. It is no proper basis to regard the term "process" elsewhere in the disclosure of Draves as limited only to application programs such as a spreadsheet program or a word processing program.

12. The term "process" as used in Draves is a generic term referring to executable software programs and it covers all processes whatever their privilege level is during execution.

13. With regard to Clifton, the Examiner determined that it discloses that even processes executed at the highest privilege levels have the potential to cause security problems and should not be permitted unrestricted access to all system resources (Answer, page 6, lines 3-17):

> Specifically, Clifton recognizes that processes executed at the highest privilege levels have the potential to cause security problems. Col. 2, lines 3-8 describes the traditional hierarchical 3-ring privilege system discussed by Appellant in the instant application (see Figure 1, Background section). Col. 1, lines 62-67 of Clifton teaches that this system can cause security problems as processes operating at highest privilege level has unlimited access to all system resources, as recognized by Appellant on page 5, paragraph 3, of the reply filed 03/18/2005. Col. 3, lines 22-25 of Clifton further note that as a result of this security problem, a single security breach at a high privilege level "often results in a complete compromise of the system", as recognized by Appellant in the first 3 lines on page 8 of the Appeal Brief filed 08/05/2005.
>
> Therefore, Clifton proposes a change to the traditional ring architecture, which applies security restrictions to even processes running at the highest privilege level. By his invention, a process which has high privilege levels is prevented from unlimited access to system resources. Col. 2, lines 38-42 teach that a process with high privilege levels may not access resources with higher clearance levels. See also Col. 2, lines 48-56, where a process with "top secret" clearance, which corresponds to a highest privilege level, is unable to access resources with "secret" clearance levels, which correspond to a lower privilege level. See also Col. 4, lines 30-35.

E. Principles of law

Obviousness is a legal determination made on the basis of underlying factual inquiries including (1) the scope and content of the prior art; (2) the differences between the claimed invention and the prior art; (3) the level of ordinary skill in the art; and (4) any objective evidence of unobviousness, Graham v. John Deere Co., 383 U.S. 1, 17, 148 USPQ 459, 467 (1966). While motivation is necessary to combine teachings, the motivation need not be expressly stated in any prior art reference. In re Kahn, 441 F.3d 977, 989, 78 USPQ2d 1329, 1338 (Fed. Cir. 2006). One with ordinary skill in the art is presumed to have skills apart from what the prior art references explicitly say. See In re Sovish, 769 F.2d 738, 743, 226 USPQ 771, 774 (Fed. Cir. 1985). There need only be an articulated reasoning with rational underpinnings to support a motivation to combine teachings. In re Kahn, 441 F.3d at 988, 78 USPQ2d at 1337. The level of ordinary skill in the art is evidenced by the applied references. See In re Oelrich,

579 F.2d 86, 91, 198 USPQ 210, 214 (CCPA 1978) ("the PTO usually must evaluate both the scope and content of the prior art and the level of ordinary skill solely on the cold words of the literature"); In re GPAC Inc., 57 F.3d 1573, 1579, 35 USPQ2d 1116, 1121 (Fed. Cir. 1995) (the Board did not err in adopting the approach that the level of skill in the art was best determined by the references of record).

F. Analysis

The Obviousness Rejection of Claims 1, 2, 6, 8-10, 14, 16-18, 22 and 24 over Draves and Clifton

Not much is in dispute between the Examiner and the applicant. The Examiner determined that the only difference between the subject matter of claims 1, 9, and 17 and the disclosure of Draves is that while claims 1, 9, and 17 require the software codes with which a first security ID is associated and one of which is actually initiated on the processor be for execution at a "most privileged level," Draves' executable processes, i.e., software codes, are not specified as being for execution at a most privileged level. (Answer, page 4, line 5 through page 5, line 4; page 8, lines 1-7.) The applicant does not challenge that finding. (Appeal Brief, page 6, line 21, through page 7, line 4.)

In the Background of the Invention portion of Draves, the discussion identifies a spreadsheet program and a word processing program as examples of two processes which may need to share a common resource (FF. 9). In the Detailed Description of Invention section and the Summary of Invention section of its disclosure, Draves refers to processes, i.e., programs, and nowhere limits a "process" desiring access to a system resource to a spreadsheet program, a word processing program, or any other type of application program (FF. 10). The reference to a spreadsheet program and a word processing program in the Background of the Invention portion

of Draves merely identifies an example of programs which may need to share resource and serves as no proper basis to read all references to "process" elsewhere in the disclosure of Draves as limited only to application type programs such as a spreadsheet program or a word processing program (FF. 11). The term "process" as used in Draves is a generic term referring to executable software and it covers all processes whatever is their privilege level during execution (FF. 12).

The Examiner cited to Clifton for its teachings that for security reasons even programs whose privilege level is the highest by conventional designation, such as an operating system program at privilege level zero, should not enjoy unrestricted access to all system resources (FF. 13). The applicant argues (Appeal Brief, page 7, line 19, through page 8, line 4) that the cited references including Draves are unconcerned with any of the problems described in the applicant's application, particularly the problem of security risks associated with executing codes or processes that are at a most privileged level. The argument is without merit, as the applicant fails to address or discuss any of the portions of Clifton cited by the Examiner for teaching that there is a security risk in executing programs which are at the highest privilege level and that even programs at the highest conventional privilege level should not be permitted unlimited access to the resources of the system. The portions cited by the examiner include those reproduced above in FF. 13 as well as a multitude of other citations by column and line number appearing on pages 13-14 of the Examiner's Answer. Not having addressed the Examiner's reasoning and cited support for determining that Clifton discloses that even programs executing at the highest privilege level can cause a security problem and therefore should not be permitted unrestricted access to all system resources, the applicant has failed to demonstrate error in that determination. We have been provided with no explanation as to why the Examiner's determination is wrong.

The applicant argues that the Examiner's conclusion of obviousness is mere conclusory. The argument is rejected. The motivation to combine teachings, need not be expressly stated in any prior art reference. In re Kahn, 441 F.3d at 989, 78 USPQ2d at 1338. The Examiner need only articulate a reasoning with rational underpinnings to support a motivation to combine teachings. In re Kahn, 441 F.3d at 988, 78 USPQ2d at 1337. Here, the Examiner has provided rational reasoning for applying Draves' general statements about providing security to processes, and for example to processes at the lowest privilege level such as spreadsheet and word processing programs, to providing security for processes at the most privileged level, and that is -- according to Clifton even processes at the most privileged level pose a security concern and should not be permitted unrestricted access to resources. On page 6 of the Answer, the Examiner first states that Clifton recognizes that processes executed at the highest privilege levels have the potential to cause security problems (lines 3-4), then states that Clifton proposes a change to the traditional ring architecture and applies security restrictions to even processes running at the highest privilege level (lines 13-14), and follows up with this paragraph (Answer, page 6, line 21, through page 7, line 2):

> One of ordinary skill in the art, having read Clifton, would easily recognize that security restrictions are needed for all processes, especially including those running at high privilege levels, and would be motivated to use the system of Draves to secure access requests for processes running at the highest privilege levels.

Finally, the applicant argues that Draves teaches away from the claimed invention (Brief, page 8, lines 5-13):

> Furthermore, Draves teaches away from the Examiner's proposed modification of the prior art. In particular, Draves teaches that the handle and the key in the resource identifier are used to provide secure access to client and/or server processes, which typically operate at the lowest privilege level (privilege

> level 3), as discussed above. Thus, Draves teaches away from associating one or more first security identifications (IDs) with each of a plurality of software codes that are to be executed by a processor at a most privileged level, as set forth in independent claims 1, 9 and 17. Accordingly, Draves also teaches away from initiating execution of one of the plurality of software codes on the processor at the most privileged level, as set forth in independent claims 1, 9, and 17.

The argument is without merit and is rejected. Draves nowhere defines its server or client processes as only processes executing at the lowest privilege level. Throughout the Summary of Invention section and the Detailed Description of the Invention section of its disclosure, and even in its Abstract, Draves consistently referred to "processes" generally and not once limited the reference to processes executing at any particular privilege level. While it is true that in the Background of the Invention section Draves does give an example of two processes which may need to share resources with each other and referred to a spreadsheet program and a word processing program (Draves, column 1, lines 23-27), no basis exists to infer therefrom that all references to "processes" in the Detailed Description of the Invention section and the Summary of the Invention section of the disclosure must necessarily be identifying a spreadsheet program or a word processing program, or a low privileged application program of the same type. The term "process" is generic and covers whatever process that is executing on the computer, whether the privilege level is 0, 1, 2, or 3. The applicant has provided no convincing rationale why what is disclosed as an example process in the Background of the Invention section of Draves should be regarded as specifically teaching that the invention can or should have no application to any other type of process. What is exemplary is not a "teaching away" from everything else. By definition, what is exemplary is not exclusionary in character. Moreover, the Detailed Description of the Invention section and the Summary of the Invention

section of Draves' disclosure remains generic insofar as references to processes are concerned and contain no reference to a spreadsheet program or a word processing program.

The Obviousness Rejections of Claims 3-5, 7, 11-13, 15, 1, and 23 over Draves, Clifton, and Holtey

Claims 3-5, 7, and 11-13 each depend from independent claim 1, 9, or 17. With regard to the rejection of these dependent claims, the applicant made no argument beyond those made with respect to the rejection of independent claims 1, 9, and 17 over the combined teachings of Draves and Clifton. Accordingly, the rejection of claims 3-5, 7, and 11-13 stand or fall with the rejection of claims 1, 9, and 17.

Conclusion

The rejection of claims 1-2, 6, 8-10, 14, 16-18, 22, and 24 as unpatentable under 35 U.S.C. § 103 over the combined teachings of Draves and Clifton is **affirmed**.

The rejection of claims 3-5, 7, 11-13, 15, 19-21, and 23 as unpatentable under 35 U.S.C. § 103 based on the combined teachings of Draves, Clifton, and Holtey is **affirmed**.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a) (1)(iv)(2005).

AFFIRMED

JAMESON LEE
Administrative Patent Judge

RICHARD TORCZON
Administrative Patent Judge

JAMES T. MOORE
Administrative Patent Judge

BOARD OF PATENT APPEALS AND INTERFERENCES

Appeal No. 2007-0564
Application 10/005,225

By First Class Mail

Attorney for the applicant:

Mark W. Sincell
Williams, Morgan & Amerson
10333 Richmond, Suite 1100
Houston, Texas 77042